Computer Science Department

TECHNICAL REPORT

BALANCING IS NOT ALWAYS GOOD

BY

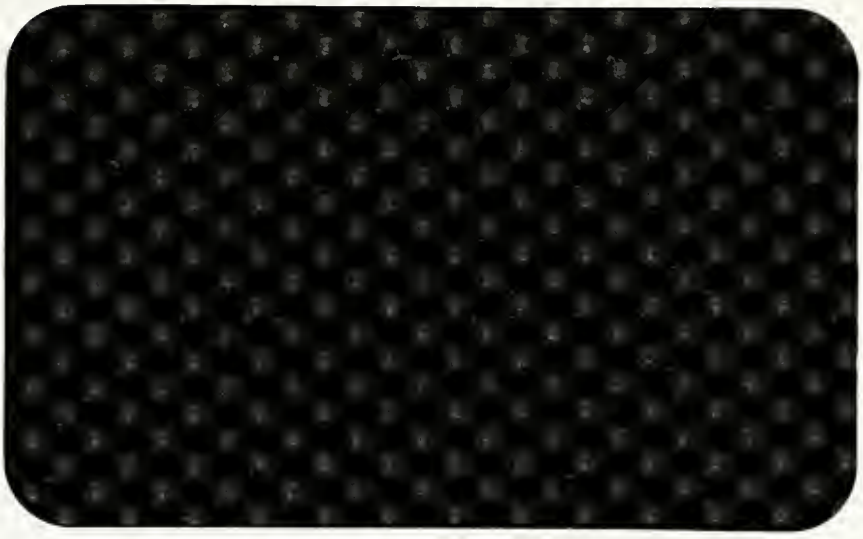
MARC SNIR

June 1981

Report No. 033

NEW YORK UNIVERSITY

Department of Computer Science
Courant Institute of Mathematical Sciences
251 MERCER STREET, NEW YORK, N.Y. 10012



BALANCING IS NOT ALWAYS GOOD

BY

MARC SNIR

June 1981

Report No. 033

This work is based upon work supported in part by the U.S. Department of Energy Contract No. DE-AC02-76ERO3077.

Abstract: Recurrence formulas arising from the analysis of divide and conquer algorithms are considered. It is shown that the balancing principle is not always valid. General solutions and error estimates are provided.

Keywords: Algorithm analysis, Divide and conquer.

1. Introduction

The time complexity of divide and conquer algorithms is often determined from a recurrence relation of the form

$$c(n) = \min_{i+j=n} c(i) + c(j) + f(n)$$
$$c(1) = f(1). \qquad (1.1)$$

c(i) and c(j) is the time required to solve two subproblems of size i and j respectively, and f(n) is the time needed to combine these two solutions. Typical examples are merge-sort (f(n)=0(n)) and maximum finding (f(n)=0(1)).

When analyzing such problems it is always assumed implicitly that an optimal solution to 1.1 is obtained by taking subproblems of equal sizes; This is the famous "Balancing Principle" [1,§2.3]. The recurrence 1.1 is therefore replaced by the recurrence

$$c(x) = c(\lfloor n/2 \rfloor) + c(\lceil n/2 \rceil) + f(n),$$
$$c(1) = f(1). \qquad (1.2)$$

The next step is to solve the functional equations

$$c(x) = 2c(x/2) + f(x),$$
$$c(1) = f(1) \qquad (1.3)$$

and use the solution as an approximation to the correct solution of 1.2.

We elaborate in this paper on each of these steps.

First, under what conditions is the Balancing Principle valid? It turns out that it is valid whenever the function f is convex. However, if f is concave an optimal solution to 1.1 is obtained by "unbalancing" as much as possible. The problem turns out to be related to that of building binary trees of minimal sum, where "tree sum" is defined to be a generalization of the external path length function.

Next, we suggest a simple analytic formula to bound the error involved in using a solution to system 1.3 as an approximation to the solution of system 1.2. Finally, we give a general solution to the system 1.3.

Our results validates the usual analysis of the recurrence relation 1.1, for convex f. When f is concave, and therefore sublinear, the solution to 1.1 is $\Theta(n)$, and (un)balancing optimally rather than evenly only affects the constant implicit in the Θ notation.

2. Optimal Splitting

Definitions: The first order difference of a function f(n) is defined to be $\Delta f(n) = \Delta^1 f(n) \stackrel{\Delta}{=} f(n+1)-f(n)$; the k-th order difference is defined as $\Delta^k f(n) \stackrel{\Delta}{=} \Delta^{k-1} f(n+1) - \Delta^{k-1} f(n)$. A function f is nondecreasing if $\Delta f \geq 0$, nonincreasing if $\Delta f \leq 0$, convex if $\Delta^2 f \geq 0$, concave if $\Delta^2 f \leq 0$. Note that if f(x) is k times differentiable and $f^{(k)} \geq 0$ then $\Delta^k f \geq 0$.

Functions f occuring in recurrence relations of the form 1.1, which describe the time complexity of divide and conquer algorithms, can be expected to be nonnegative and nondecreasing. Generally, they also will be convex, by a law of diminishing returns: the amount of work required to handle and extra item does not decrease as the number of items increases. There are exceptions to this rule however: for an example see [2].

The optimal splitting of n, that yields the solution to (1.1), is given by the next two theorems.

THEOREM 2.1. Let c be defined by the recurrence relation 1.1. Assume that f is nonnegative, nondecreasing and convex

($f \geq 0$, $\Delta f \geq 0$ and $\Delta^2 f \geq 0$). Then for $n > 1$.

$$c(n) = c(\lfloor n/2 \rfloor) + c(\lceil n/2 \rceil) + f(n). \qquad (2.1)$$

PROOF: By induction on n. We shall also prove the auxilary assertion

$$\Delta^2 c(n-2) \geq 0$$

Both assertions are valid for $n \leq 4$. Assume that both are valid for all $n' < n$. Let

$$c(n) = c(i) + c(j) + f(n)$$

with $i + j = n$, $0 < i \leq j < n$. Assume that $i + 1 \leq j - 1$. Then

$$\begin{aligned} & c(n) - [c(i+1) + c(j-1) + f(n)] \\ &= c(i) - c(i+1) + c(j) - c(j-1) \\ &= \Delta c(j-1) - \Delta c(i) \geq 0, \end{aligned}$$

by the second inductive assertion. It follows, by the definition of c, that

$$c(n) = c(i+1) + c(j-1) + f(n).$$

A repeated use of this implication proves equality 2.1. As for the second assertion, if $n=2k$ then

$$\begin{aligned} \Delta^2 c(n-2) &= c(n) - 2c(n-1) + c(n-2) \\ &= [2c(k) + f(2k)] - 2[c(k) + c(k-1) + f(2k-1)] + \\ &\quad + [2c(k-1) + f(2k-2)] \\ &= f(2k) - 2f(2k-1) + f(2k-2) \\ &= \Delta^2 f(n) \\ &\geq 0. \end{aligned}$$

The proof for $n=2k+1$ is similar.

□

Note that if $\Delta^2 f>0$ (f is strictly convex) then all the inequalities in the last proof are tight, and the decomposition that achieves the minimum in 1.1 is unique.

The optimal strategy when f is concave is more surprising:

THOEREM 2.2. Let c be defined by the recurrence relation 1.1. Assume that f is nonnegative, nondecreasing and concave ($f\geq 0$, $\Delta f\geq 0$ and $\Delta^2 f\leq 0$). If $3\cdot 2^{k-1}\leq n\leq 3\cdot 2^k$ then

$$c(n) = c(2^k) + c(n-2^k) + f(n).$$

The proof we currently have for this theorem is a lengthy, uninstructive case analysis, and we have relegated it to an appendix.

□

The last two theorems are better understood if we use a binary tree representation for the recurrence relation 1.1.

Let us define w(T), the weight of a binary tree T, to be the number of leaves in T, and let the f-sum of T be defined as

$$\Sigma_f(T) \triangleq \sum_{T' \text{ subtree of } T} f(w(T')).$$

Note that the external path length of the tree T is equal to $\Sigma_{id}(T)$, where id is the identity function.

The f-sum of T can be defined recursively:

$\Sigma_f(T) = 0$ if T is empty;

$\Sigma_f(T) = \Sigma_f(T_l) + \Sigma_f(T_r) + f(w(T))$,

where T_l and T_r are the left and right subtrees of T, otherwise.

It follows that $\Sigma_f(n)$, the minimal f-sum of a binary tree with n leaves, fulfills the recurrence relation

$$\Sigma_f(1) = f(1)$$

$$\Sigma_f(n) = \min_{i+j=n} \Sigma_f(i) + \Sigma_f(j) + f(n).$$

We have proven that

THEOREM 2.3. If c is the solution to the recurrence relation 1.1 then c(n) is equal to the minimal f-sum of a binary tree with n leaves.

Theorems 2.1 and 2.2 can now be seen to be assertions about the structure of minimal f-sum trees. Clearly, minimal f-sum trees are extended binary trees [3, §2.3,4.5]: each node has either two children or none.

A binary tree is said to be (weight) balanced if its left and right subtrees are balanced, and their weight differ at most by one. Note that this definition is different (stricter) than that usually given in the literature. We can now restate theorem 2.1:

THEOREM 2.1´. If f is nonnegative, nondecreasing, and convex, then balanced trees have minimal f-sum; if f is strictly convex, then a tree has minimal f-sum iff it is balanced.

A balanced tree with 10 leaves is shown in Figure 1; Figure 2 shows a tree with 10 leaves that fits the conditions of theorem 2.2. Before discussing it, we need the following definitions.

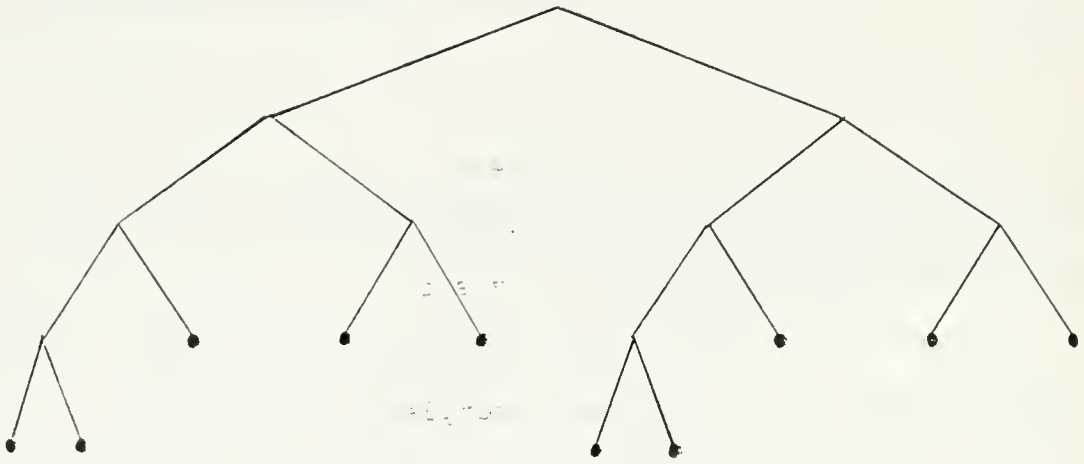


Figure 1
Balanced tree with 10 leaves

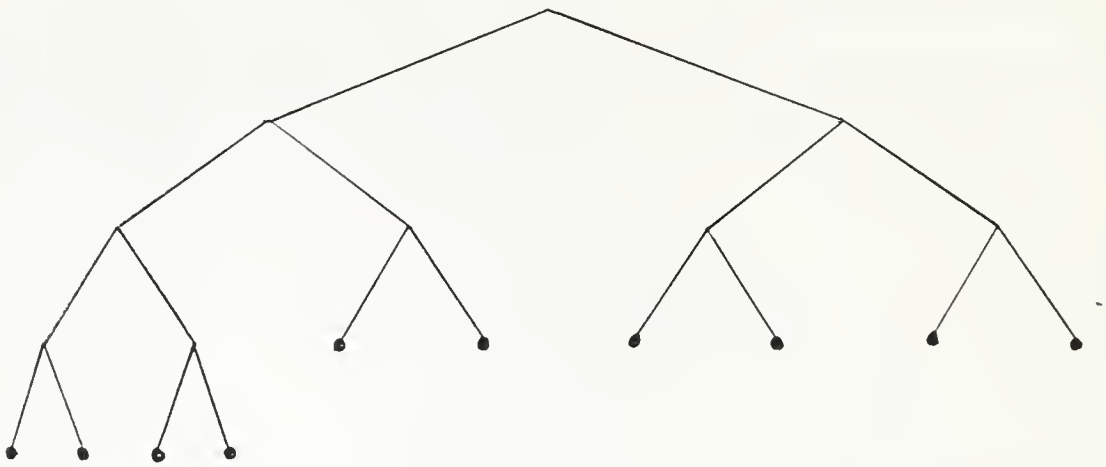


Figure 2
Heap tree with 10 leaves

The level of a node v in a tree is the length of the path from the root to v; h(T), the height of the tree T, is the maximum level of a node in it. Note that $w(T) \leq 2^{h(T)}$. An extended binary tree T is complete if all its leaves occur at the last two levels only. Note that a complete extended binary tree with n leaves has height $h(T) = \lceil \lg n \rceil$; $2^{\lceil \lg n \rceil} - n$ of the leaves occur at level $h(T)-1$ and $2n - 2^{\lceil \lg n \rceil}$ of the leaves occur at level h(T) (cf. [3, §5.3.1]).

A recursive definition of a complete tree can be given as follows:

LEMMA 2.4. An extended binary tree T is complete iff

$w(T) \leq 1$ or

The left and right subtrees T_l and T_r of T are complete and

$$2^{\lfloor \lg n \rfloor - 1} \leq w(T_l), w(T_r) \leq 2^{\lceil \lg n \rceil - 1}, \qquad (2.3)$$

where n=w(T).

PROOF: If T is complete, then T_l and T_r have all their leaves at levels $\lceil \lg n \rceil - 1$ and $\lfloor \lg n \rfloor - 1$. It follows that both subtrees are complete and that the number of leaves in each of them fulfills inequality 2.3.

Conversely, assume that inequality 2.3 is fulfilled, and that T_l and T_r are complete. Then all the leaves in T_l and T_r are at levels $\lfloor \lg n \rfloor - 1$ and $\lceil \lg n \rceil - 1$. It follows that T is complete.

□

Note that a balanced tree is complete, by the last lemma.

The tree illustrated in figure 2 corresponds to the usual image of a complete tree: its nodes can be numbered from 1 to 2n-1, such that the parent of a node k is the node $\lfloor k/2 \rfloor$ (the root has number 1, and has no parent). We define a heap tree to be a tree for which such labelling exists. Alternative definitions of a heap tree are given in the following theorem.

Theorem 2.5. Let T be an extended binary tree. Then the following three conditions are equivalent.

1. T is a heap tree.
2. T is a maximally unbalanced complete tree, e.g. T is complete, and at each node the difference between the weights of the left and right subtree is maximum.
3. For each subtree T' of T, if $3\cdot 2^{k-1} \le w(T') \le 3\cdot 2^k$, then T' has left and right subtrees with weights 2^k and $w(T')-2^k$.

PROOF: 1=>3. If T is a heap tree and T_l and T_r are its left and right subtrees, then either T_r is full and has 2^k leaves, where $2^k \le w(T_l) \le 2^{k+1}$, or T_l is full and has 2^k leaves, where $2^{k-1} \le w(T_r) \le 2^k$. In both cases assertion 3 is valid for T itself. Since each subtree of a heap tree is a heap tree, the implication follows also for each subtree.

3=>2. Let $n=w(T')$. Note first that $2^{\lfloor \lg n \rfloor - 1} \le 2^k$, $n-2^k \le 2^{\lceil \lg n \rceil - 1}$, so that, by lemma 2.4, T is a complete tree. It is easy to check that if $3\cdot 2^{k-1} \le n \le 3\cdot 2^k$ then $|i-j|$ is maximum under the constraints

$$i+j=n$$
$$2^{\lfloor \lg n \rfloor - 1} \le i,j \le 2^{\lceil \lg n \rceil - 1},$$

when $i=2^k$, $j=n-2^k$ (or vice-versa). Thus, condition 3 ensures that T is a maximally unbalanced complete tree.

2=>1. Assume w.l.o.g. that at each node of T the left

subtree is no smaller than the right subtree. Number the nodes of T from level 0 to level h(d), and at each level, from left to right. If some node k has no children, whereas a node k′, k′>k, has two children then one can obtain a more unbalanced tree that is still complete by moving the children of k′ to k. It follows that T is a heap tree.

□

Note that condition 2 of the last theorem is exactly the condition stated in theorem 2.2. We can therefore restate this theorem:

THEOREM 2.2′: <u>If f is nonnegative, nondecreasing and concave, then heap trees have minimal f-sum; if f is strictly concave then a tree has minimal f-sum iff it is a heap tree.</u>

If $\Delta^2 f=0$, then both theorems 2.1′ and 2.2′ are valid. Indeed, if $\Delta^2 f=0$ then Σ_f is a linear function of the external path length. But all complete trees have minimal external path length, including the most balanced and the most unbalanced ones.

3. Error estimates

A simple estimate on the error involved in solving recurrence 1.3 with rational domain rather than recurrence 1.2 with integer domain is given by the following theorem.

THEOREM 3.1. <u>Let c be defined by</u>

$$c(1) = f(1)$$

$$c(n) = c(\lfloor n/2 \rfloor) + c(\lceil n/2 \rceil) + f(n)$$

<u>Let $\bar{c}(x)$ be the solution to the recurrence relation</u>

$$\bar{c}(1) = f(1)$$

$$\bar{c}(x) = 2\bar{c}(x/2) + f(x)$$

Let $K = \max_{1\le x\le n} |\bar{c}''(x)|$. Then

$$|c(n) - \bar{c}(n)| \le (n-1)K/2.$$

PROOF: By induction on n. For n=1 $c(1) - \bar{c}(1) = 0$. Assume that 3.1 is valid for all $n'<n$. If n=2m then

$$|c(2m) - \bar{c}(2m)| = |2c(m) - 2\bar{c}(m)| \le$$

$$\le 2(m-1)K/2 < (2m-1)K/2$$

If n=2m+1 then

$$|c(2m+1) - \bar{c}(2m+1)| = |c(m) + c(m+1) - 2\bar{c}(m+\tfrac{1}{2})| \le$$

$$\le |c(m) - \bar{c}(m)| + |c(m+1) - \bar{c}(m+1)| + |\bar{c}(m) + \bar{c}(m+1) - 2\bar{c}(m+\tfrac{1}{2})|$$

But if g is twice differenciable then

$$g(x+\delta) + g(x-\delta) - 2g(x) = 2\delta^2 g''(\xi)$$

for some $x-\delta\le\xi\le x+\delta$. Therefore,

$$|c(2m+1) - \bar{c}(2m+1)| \le |c(m) - \bar{c}(m)| + |c(m+1) - \bar{c}(m+1)| + \tfrac{1}{2}K$$

$$\le (m-1)K/2 + mK/2 + K/2 = (2m)K/2.$$

□

4. General solution

The recurrence 1.3 can be transformed into a linear recurrence by a change of variables, and solved by standard methods (see [4]). We obtain that the solution of the system

$$c(1) = 1$$

$$c(x) = 2c(x/2) + x^d$$

is given by

$$c(x) = \begin{cases} 2x-1 & \text{if } d=0 \\ x\lg x + x & \text{if } d=1 \\ (2^d/(2^d-1))(x^d - x) & \text{if } d>1. \end{cases}$$

Thus, if $f(x) = \Sigma a_i x^i$, the solution to the recurrence 1.3 is given by

$$c(x) = a_0(2x-1)+a_1(x\lg x+x) + \sum_{i\geq 1} a_i(2^{i-1}/(2^{i-1}-1))(x^i-x) =$$

$$= -a_0+(2a_0+a_1-\sum_{i\geq 1}(2^{i-1}/(2^{i-1}-1))a_i)x + \sum_{i>1} a_i(2^{i-1}/(2^{i-1}-1))x^i+a_1x\lg x.$$

In particular, if f is a polynomial of degree d, $f(x) = \sum_{i=0}^{d} a_i x_i$, then

$$c(x) = -a_0+(2a_0+a_1-\sum_{i=2}^{d}(2^{i-1}/(2^{i-1}-1))a_i)x + \sum_{i=2}^{d} a_i(2^{i-1}/(2^{i-1}-1))x^i+a_1x\lg x \quad (4.1)$$

Assume now that $f(x) = \sum_{i=0}^{d} a_i x^i$, and that $f\geq 0$, $\Delta f\geq 0$ and $\Delta^2 f\geq 0$, so that the conditions of theorem 2.1 are fulfilled. The function c(n), defined by 4.1, is an approximation to the solution of 1.1, with an error bounded, according to theorem 3.1, by $n\cdot \max_{1\geq x\geq n} c''(n) = 0(n^{d-1})$. We have therefore that the solution to the recurrence relation 1.1 is of the form

$$c(n) = \begin{cases} a_d(2^{d-1}/(2^{d-1}-1))n^d + 0(n^{d-1}), & \text{if } d>1 \\ a_1 n(\lg n+1) + 0(1), & \text{if } d=1 \\ a_0(2n-1), & \text{if } d=0. \end{cases} \quad (4.2)$$

Now, if the leading coefficient of f is positive,

then f and its first two differences will be nonnegative for x large enough. It is easy to check that the error involved in using the function defined by 3.1 as a solution to the recurrence 1.1 is still restricted to the lower order terms. We can obtain the following result.

THEOREM 4.1. The solution to the recurrence relation

$$c(1) = 0(1)$$

$$c(n) = \min_{i+j=n} c(i)+c(j) + an^d + 0(n^{d-1}),$$

where $a>0$, is given by

$$c(n) = \begin{cases} a(2^{d-1}/(2^{d-1}-1))n^d+0(n^{d-1}) & \text{if } d>1 \\ an\lg n + 0(n) & \text{if } d=1. \end{cases}$$

REFERENCES

1. A. AHO, J. HOPCROFT and J. ULLMAN, The Design and Analysis of Computer Algorithms, Addison-Wesley, Reading, Mass., 1974.

2. J. BENTLEY and M. SHAMOS, Divide and Conquer for Linear Expected Time. Information Processing Letters 7, 87-91, 1978.

3. D.E. KNUTH, The Art of Computer Programming, Vol. 1 - Fundamental Algorithms, Vol. 3 - Sorting and Searching. Addison-Wesley, Reading, Mass., 1973.

4. G. LUEKER, Some Techniques for Solving Recurrences. ACM Computing Surveys 12, 419-436, 1980.

Appendix

Proof of theorem 2.2: By induction on n: We have to prove that if $3\cdot 2^{k-1} \le n \le 3\cdot 2^{k}$ then

$$c(n) = c(2^k) + c(n-2^k) + f(n) \qquad (A.1)$$

We assume that this equality is valid for all n', $n'<n$.

Let $n=n^k+m$, so that $2^{k-1} \le m \le 2^{k+1}$. We shall prove that for every j

$$c(2^k) + c(m) \le c(2^k+j) + c(m-j) \qquad (A.2)$$

We know that this inequality is valid for all k', $k'<k$. We also have that if $i<j$ then

$$f(i) \le f(j) \qquad (A.3)$$

and, since f is concave, if $i<j$ and $0\le r\le (j-i)/2$ then

$$f(i) + f(j) \le f(i+r) + f(j-r) \qquad (A.4)$$

We proceed with a proof by cases.

1. $m\le 2^k$. For symmetry reasons we may restrict ourselves to values of j such that $(m-2^k)/2 \le j < m$

1.1. $j>2^{k-1}$. Then

$$\begin{aligned} & c(2^k+j) + c(m-j) \\ &= c(2^k)+c(j)+f(2^k+j)+c(m-j) && \text{(By A.1)} \\ &\ge c(2^k)+c(j)+c(m-j)+f(m) && \text{(By A.3)} \\ &\ge c(2^k)+c(m). \end{aligned}$$

1.2. $0<j\le 2^{k-1}$. Then

$$\begin{aligned} & c(2^k+j)+c(m-j) \\ &= c(2^{k-1})+c(2^{k-1}+j)+f(2^k+j)+c(m-j) && \text{(By A.1)} \\ &\ge c(2^{k-1}+j)+c(m-j)+f(2^k)+c(2^{k-1}) && \text{(By A.3)} \\ &\ge c(2^{k-1})+c(m)+f(2^k)+c(2^{k-1}) && \text{(By A.2)} \\ &= c(2^k)+c(m). \end{aligned}$$

1.3. $j<0$

1.3.1. $m-j \geq 3 \cdot 2^{k-2}$. Then

$$c(2^k+j)+c(m-j)$$
$$= c(2^{k-1})+c(2^{k-1}+j)+f(2^k+j)+c(2^{k-1})+c(m-j-2^{k-1})+f(m-j) \qquad \text{(By A.1)}$$
$$\geq c(2^{k-1})+c(2^{k-1})+f(2^k)+c(2^{k-1}+j)+c(m-j-2^{k-1})+f(m) \qquad \text{(By A.4)}$$
$$\geq c(2^k)+c(m).$$

1.3.2. $m-j \leq 3 \cdot 2^{k-2}$. Then

$$c(2^k+j)+c(m-j)$$
$$= c(2^{k-1})+c(2^{k-1}+j)+f(2^k+j)+c(2^{k-2})+c(m+j-2^{k-2})+f(m-j) \qquad \text{(By A.1)}$$
$$\geq c(2^{k-1})+c(2^{k-1})+f(2^k+j)+c(2^{k-2})+c(m-2^{k-2})+f(m-j) \qquad \text{(By A.2)}$$
$$\geq c(2^{k-1})+c(2^{k-1})+f(2^k)+c(2^{k-2})+c(m-2^{k-2})+f(m) \qquad \text{(By A.4)}$$
$$\geq c(2^k)+c(m).$$

2. $m>2^k$. We have to consider values of j such that $-2^k \leq j \leq (m-2^k)/2$.

2.1. $j>0$. Note that $j \leq 2^{k-1}$

2.1.1. $m-j \leq 3 \cdot 2^{k-1}$. Then

$$c(2^k+j)+c(m-j)$$
$$= c(2^{k-1})+c(2^{k-1}+j)+f(2^k+j)+c(2^{k-1})+c(m-j-2^{k-1})+f(m-j) \qquad \text{(By A.1)}$$
$$\geq c(2^{k-1})+c(2^{k-1})+f(2^k+j)+c(2^{k-1})+c(m-2^{k-1})+f(m-j) \qquad \text{(By A.2)}$$
$$\geq c(2^{k-1})+c(2^{k-1})+f(2^k)+c(2^{k-1})+c(m-2^{k-1})+f(m) \qquad \text{(By A.4)}$$
$$\geq c(2^k)+c(m).$$

2.1.2. $m-j>3 \cdot 2^{k-1}$. Then $m-2^{k-1} \geq 3 \cdot 2^{k-2}$ and

$$c(2^k+j)+c(m-j)$$
$$= c(2^{k-1})+c(2^{k-1}+j)+f(2^k+j)+c(2^k)+c(m-j-2^k)+f(m-j) \qquad \text{(By A.1)}$$
$$\geq c(2^{k-1})+c(2^{k-1})+f(2^k+j)+c(2^k)+c(m-2^k)+f(m-j) \qquad \text{(By A.2)}$$
$$\geq c(2^{k-1})+c(2^{k-1})+f(2^k)+c(2^k)+c(m-2^k)+f(m) \qquad \text{(By A.4)}$$
$$\geq c(2^k)+c(m).$$

2.2 $j<0$.

2.2.1. $m-j \geq 3 \cdot 2^{k-1}$. Then

$$c(2^k+j)+c(m-j)$$
$$= c(2^k+j)+c(2^k)+c(m-j-2^k)+f(m-j) \qquad \text{(By A.1)}$$
$$\geq c(2^k)+c(2^k+j)+c(m-j-2^k)+f(m) \qquad \text{(By A.3)}$$
$$\geq c(2^k)+c(m)$$

2.2.2. $m-j < 3 \cdot 2^{k-1}$. Note that $m+2^{k-1} \leq 3 \cdot 2^k$, so that

$$c(2^k+j)+c(m-j)$$
$$= c(2^k+j)+c(2^{k-1})+c(m-j-2^{k-1})+f(m-j) \qquad \text{(By A.1)}$$
$$\geq c(2^{k-1})+c(2^k+j)+c(m-j-2^{k-1})+f(m) \qquad \text{(By A.3)}$$
$$\geq c(2^{k-1})+c(2^k)+c(m-2^{k-1})+f(m) \qquad \text{(By A.2)}$$
$$\geq c(2^k)+c(m).$$

□

Note again that if f is strictly concave ($\Delta^2 f < 0$) then the decomposition corresponding to a solution of 1.1 is unique.